CLAIMS

What is claimed is:

1. A method for creating a superfingerprint for identifying a software comprising:

executing said software at least once;

in each execution, selecting specified portions of at least one of said executing software and of results of executing said software;

in each execution, performing computations on said selected portions to obtain a collection of fingerprints; and

combining said collections of fingerprints found in each execution into the superfingerprint of said software according to a combining rule.

2. The method of claim 1 wherein the software is executed a plurality of times and the collection of fingerprints obtained during each execution are combined together according to the combining rule.

3. The method of claim 2 wherein the combining rule outputs only those fingerprints that are computed in more than a specified number of executions.

4. The method of claim 1 wherein the combining rule removes from the output those fingerprints that occur in more than a specified number of executions of specified other softwares.

5. The method of claim 4 wherein fingerprints are not removed if they belong to a same group of software as said software.

6. The method of claim 1 wherein a fingerprint belongs to the superfingerprints of several softwares.

7. The method of claim 6 further comprising:

storing in at least one data structure at least one fingerprint and means to identify said several softwares in whose superfingerprint said fingerprint is included.

8. The method of claim 7 wherein the means to identify is a bit vector data structure whose mth bit indicates whether the superfingerprint associated with the mth member of said several softwares includes said fingerprint.

9. The method of claim 7 wherein associated with each said fingerprint there are at least two numbers k1 and k2 where k2 is greater than or equal to k1 that indicate that the superfingerprints of softwares from k1 to k2 of said several softwares all include said fingerprint.

10. The method of claim 6 wherein said several softwares belong to a group of software.

11. The method of claim 1 wherein the fingerprints of various softwares are stored in a data structure to facilitate and accelerate retrieval of fingerprints and associated names of software.

12. The method of claim 1 wherein the executing software is partitioned into pages, said specified portions are selected from said pages and the computations produce a fingerprint for each portion.

13. The method of claim 1 wherein said specified portions are selected from the software stored in a memory of the device executing said software.

14. The method of claim 1 wherein said specified portions are selected from the software stored in secondary memory of the device executing said software.

15. The method of claim 1 wherein the specified portions are basic blocks of programs.

16. The method of claim 1 wherein the computation involves only parts of said selected portions.

17. The method of claim 16 wherein said involved parts are operation codes.

18. The method of claim 16 wherein said involved parts are information in a audio signal.

19. The method of claim 16 wherein said involved parts are information in a visual display.

20. The method of claim 1 wherein the selected portion concerns the interaction between at least one user and the execution of software.

21. The method of claim 1 wherein the input to the computation is a sequence.

22. The method of claim 1 wherein the computation is a hash function value of said portion.

23. The method of claim 22 wherein the hash function value is computed by polynomial fingerprinting.

24. The method of claim 1 wherein the computation is a computation on an audio signal.

25. The method of claim 1 wherein the computation is a computation on a video stream.

26. A method for identifying a first software comprising the steps of:

storing previously created superfingerprints for at least one software;

executing said first software at least once;

selecting specified portions of at least one of said executing software and of the results of executing said software on each execution;

performing specified computations on said selected portions to obtain a collection of fingerprints;

comparing said collection of fingerprints to said previously computed superfingerprint of at least one second software to determine whether there is an approximate match; and

declaring said first software to be the same as said second software if an approximate match is found.

27. The method of claim 26 wherein said specified portions of said executing software and of said results of executing said software, are stored in a memory of a device executing said software.

28. The method of claim 27 wherein said specified portions of at least one of said executing software and of said results of executing said software, are selected from recently accessed portions of the software stored in the memory of the device executing said software.

29. The method of claim 27 wherein said specified components of said executing software are selected from portions of said executing software stored in secondary storage.

30. The method of claim 26 wherein the portions of said executing software are selected while said software is sent from one device to another.

31. The method of claim 26 wherein said portions of the results of execution of said software are selected from the output of the device executing said software.

32. The method of claim 26 wherein said specified portions of at least one of said executing software and of the results of executing said software on a later execution are dependent on the results of an earlier approximate match.

33. The method of claim 26 wherein determining the approximate match comprises: determining whether the amount of said commonality between said fingerprints of said first software and the fingerprints comprising said superfingerprint of said at least one second software exceeds a specified threshold in which case the first software is identified to be the same as the second software.

34. The method of claim 33 wherein said specified threshold is exceeded only if the amount of commonality between said fingerprints of said first software and the fingerprints comprising said superfingerprint of said second software exceed the commonality between said fingerprints of said first software and the fingerprints comprising the superfingerprint of a third software.

35. The method of claim 33 wherein the commonality between the fingerprints of said first software and said second software depends on the number of fingerprints that are the same in said two softwares with a weighting factor for each equal fingerprint.

36. The method of claim 35 wherein commonality further depends on the number of fingerprints that are different in said two softwares with a weighting factor for each unequal fingerprint.

37. The method of claim 35 wherein commonality further depends on the relative positions of the portions of Software from which at least two fingerprints are computed.

38. The method of claim 26 wherein the specified computation involves only parts of said selected portions.

39. The method of claim 38 wherein said involved parts are operation codes.

40. The method of claim 38 wherein said involved parts are information in an audio signal.

41. The method of claim 38 wherein said involved parts are information in a visual display.

42. The method of claim 26 wherein the selected portion concerns the interaction between at least one user and the execution of software.

43. The method of claim 26 wherein the input to the computation is a sequence.

44. The method of claim 26 wherein the input to the comparison is a collection of fingerprints each having an associated weight.

45. The method of claim 26 wherein the computation is a hash function value of said portion.

46. The method of claim 45 wherein the hash function value is computed by polynomial fingerprinting.

47. The method of claim 26 wherein the computation is a computation on an audio signal.

48. The method of claim 26 wherein the computation is a computation on a video stream.

49. A method for identifying a software group of a first software comprising the steps of:

storing previously created superfingerprints for at least one software group;

executing said first software at least once;

selecting specified portions of said executing first software and of the results of executing said first software on each execution;

performing specified computations on said selected portions to obtain a collection of fingerprints;

comparing said collection of fingerprints to said previously computed superfingerprint of at least one second software group to determine whether there is an approximate match; and

declaring said first software to be a member of said second software group if an approximate match is found.

50. A method for identifying a software that is a member of a group of software comprising the steps of

storing previously created superfingerprints for at least one software group and for members of that group;

executing said software at least once;

selecting specified portions of said executing software and of the results of executing said software on each execution;

performing computations on said selected portions to obtain a collection of fingerprints;

comparing said collection of fingerprints to said previously computed superfingerprint of at least one second software group and the superfingerprints of the members of said second software group to determine whether there is an approximate match with said group and at least one of said superfingerprints of said members of said group; and

declaring said software to be the same as a particular member of said second software group if an approximate match is found.